Create a calculator to work with rational numbers.

Requirements:

○ It should provide capability to add, subtract, divide and multiply rational

numbers

○ Create a method to compute GCD (this will come in handy during operations on

rational)

Add option to work with whole numbers which are also rational numbers i.e. (n/1)

- achieve the above using auxiliary constructors

- enable method overloading to enable each function to work with numbers and rational.

Program:

**object** Main

{

**def** main(args: Array[*String*])

{

**val** first=**new** Rational(1,300)

**val** second=**new** Rational(4,1500)

println("Adding first and second "+(first+second))

println("subtract first and second "+(first-second))

println("multiply first and second "+(first\*second))

println("divide first and second "+(first/second))

println("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

**val** third =**new** Rational(6,1)

println("Adding first and third "+(first+third))

println("subtract first and third "+(first-third))

println("multiply first and third "+(first\*third))

println("divide first and third "+(first/third))

}

}

**class** Rational(n: Int, d: Int)

{

require(d != 0)

**private** **val** g = gcd(n.abs, d.abs)

**val** number = n / g

**val** denom = d / g

**def** **this**(n: Int) = **this**(n, 1) // auxiliary constructor

**def** + (that: Rational): Rational =

**new** Rational(

number \* that.denom + that.number \* denom,

denom \* that.denom

)

**def** + (i: Int): Rational =

**new** Rational(number + i \* denom, denom)

**def** - (that: Rational): Rational =

**new** Rational(

number \* that.denom - that.number \* denom,

denom \* that.denom

)

**def** - (i: Int): Rational =

**new** Rational(number - i \* denom, denom)

**def** \* (that: Rational): Rational =

**new** Rational(number \* that.number, denom \* that.denom)

**def** \* (i: Int): Rational =

**new** Rational(number \* i, denom)

**def** / (that: Rational): Rational =

**new** Rational(number \* that.denom, denom \* that.number)

**def** / (i: Int): Rational =

**new** Rational(number, denom \* i)

**override** **def** toString = number + "/" + denom

**private** **def** gcd(a: Int, b: Int): Int =

**if** (b == 0) a **else** gcd(b, a % b)

}

o/p:

